REMARKS

Claims 1, 3, 4, 6-14, 16, 17, 19, 21-25, 27-35, and 37-39 are currently pending in the subject application and are presently under consideration. Claims 1, 13, 22 and 34 have been amended as shown on pp. 2-8 of the Reply.

Applicants' representative thanks the Examiner for the courtesies extended during the teleconference of February 2, 2009.

Favorable reconsideration of the subject patent application is respectfully requested in view of the comments and amendments herein.

I. Rejection of Claims 1, 3, 4, 6-14, 16, 17, 21-25, 27-32, 34, 35, 37, and 38 Under 35 U.S.C. §101

Claims 1, 3, 4, 6-14, 16, 17, 21-25, 27-32, 34, 35, 37, and 38 stand rejected under 35 U.S.C. §101 because the claimed invention is directed to non-statutory subject matter. Independent claim 1 has been amended herein to clearly illustrate that elements within such claims are components associated with a computer processor. In particular, claim 1 as amended is directed towards a parallel software execution system, comprising ***a computer processor coupled to a memory for executing the following***, a first data structure, a second data structure, and a third data structure. (Support for these amendments can be found on pg. 6, lines 8-15 and pg. 23, lines 20-26). Accordingly, this claim includes functional descriptive material within a computer processor coupled to a memory, thereby rendering it structurally and functionally interrelated to the computer processor and memory, and is therefore directed to statutory subject matter. Accordingly, this rejection should be withdrawn with regard to claims 1, 3, 4, 6-14, 16, 17 and 21.

Claim 22 has been amended to illustrate that the series of steps to be performed are tied to a particular apparatus, a processor and a memory. In particular, claim 22 as amended is directed towards a method of producing an object schema, comprising: ***employing a processor coupled to a memory to execute the production of the object schema,*** ... (Support for these amendments can be found on pg. 6, lines 8-15 and pg. 23, lines 20-26). Accordingly, this claim includes functional descriptive material within a computer processor and memory, thereby tying the process to another statutory category that accomplishes the claimed method steps and is therefore directed to statutory subject matter. Further, claim 34 has been similarly amended.

Accordingly, this rejection should be withdrawn with regard to claims 22-25, 27-32, 34, 35, 37 and 38.

II. Rejection of Claims 1, 3-4, 6-12, 22-25, 27-35, and 37-39 Under 35 U.S.C. §103(a)

Claims 1, 3-4, 6-12, 22-25, 27-35, and 37-39 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Wotring *et al.* (US 6,853,997) in view of Wang *et al.* (US 6,907,433), further in view of Ludwig *et al.* (US 6,006,230), and further in view of Chua (US 2005/0210124). It is respectfully submitted that this rejection should be withdrawn for the following reasons. Wotring *et al.*, Wang *et al.*, Ludwig *et al.* and Chua, individually or in combination, do not teach or suggest each and every element set forth in the subject claims.

Applicants' claimed subject matter relates to a system and method to facilitate mapping between disparate domain models such as an object oriented programming model and a relational database model. Specifically, the system discloses an object schema component that provides a mechanism for providing a bridge between a transient graph of objects and a related persistent data store. Independent claim 1 recites, namely: a computer executable data structure comprising: ... *a third data structure that describes relationships between objects and persists object data to a database, wherein an object schema that describes data classes as well as relations between the data classes as specified in an object oriented model, is generated and utilized together with a relational schema and a mapping schema to map the programmatic objects to tables in the database; wherein the mapping schema provides the mapping between the object schema and the relational schema, and* ***the relational schema utilizes metadata associated with the database to generate an implementation neutral format or an implementation specific format that represents the database structure****; wherein the member properties include an alias attribute that is employed by a query language to identify a private member used to generate a query, the alias points to a public member that is to be utilized in place of the associated private member in text of a query; and* ***wherein the member properties include a hidden attribute that defines if there is a hidden member in a corresponding class and manages the hidden member in a transparent fashion***. The cited references fail to teach or suggest such aspects of the claimed subject matter.

Wotring *et al.* discloses a system and method for allowing data to be shared without requiring that the data be remodeled to fit a common format or convention. The users of the data

may keep their own data formats and may dynamically transform the data contained in their structure into a structure compatible with another definition without having to physically change their data or its structure. Specifically, the data is transformed from a Relational Database Management System (RDBMS) to a hierarchical format. Since RDBMS information is stored in separate tables joined through a specified key structure, information needs to be repackaged as a whole for use in data communication across a local area network (LAN), or a wide area network (WAN). The system discloses transforming data stored in relational format into a hierarchical format, such as a markup language. As information is transformed into the hierarchical structure from a RDMBS, the information then assumes the hierarchical representation of the logical records contained in the database. (*See* col. 2, line 63-col. 3, line 52).

In contrast, applicants' claimed subject matter discloses a system for mapping object components to relational components. The object schema of the system provides one portion of a three-part mapping. The other two schema components are a mapping schema and a relational schema. The object schema component describes data classes as well as relations there between as specified in an object-oriented model, for example. Relational schema component utilizes metadata associated with a database to generate an implementation neutral or implementation specific format that represents the precise database structure and data. Mapping schema component provides the mapping between the object schema and the relational schema. (*See* pg. 9, line 30-pg. 10, line 26). Furthermore, members associated with each class to be persisted are identified. Members can include class fields and properties. Members can be compound members comprising at least one field or property and another compound member. Thus, a compound member can be an array. Furthermore, it should be appreciated that member attributes can also be specified. For example, a member can be identified as a key or a member can identify an alias. (*See* pg. 22, lines 3-13).

Whereas, Wotring *et al.* merely discloses transforming data stored in relational format into a hierarchical format such as a markup language. The system provides for mapping each of the plurality of elements in the hierarchical data entity to information in a relational dataset, and transforming the relational dataset into corresponding mapped elements in the hierarchical data entity to form a hierarchical data structure. Wotring *et al.* does not disclose utilizing metadata associated with a database to generate an implementation neutral or implementation specific format that represents the precise database structure and data, as in applicants' claimed subject

matter. Specifically, applicants' claimed subject matter defines the object schema in a declarative manner such that it is specified as information external to programming logic. By providing this information outside of a users type definitions, the schema can be deployed independently of an associated application thereby allowing persistence storage of objects to change without forcing a user to recompile and redeploy application code. Wotring *et al.* discloses mapping elements to form a hierarchical data structure, and does not utilize metadata to generate an implementation neutral or implementation specific format.

Wang *et al.* does not make up for the aforementioned deficiencies of Wotring *et al.* with respect to independent claim 1. Wang *et al.* discloses a system for managing object to relational, one-to-many mapping. The system uses mapping of meta-data to generate instructions to manipulate target objects and relationships in a relational database. The mapping of meta-data contains information as to how object classes of the object model map to tables in the database and how relationships map to foreign keys. (*See* col. 2, lines 3-25).

As stated *supra*, applicants' claimed subject matter discloses a system for mapping object components to relational components that describes data classes. The members associated with each class to be persisted are identified. Members can include class fields and properties. Members can be compound members comprising at least one field or property and another compound member. Thus, a compound member can be an array. Furthermore, it should be appreciated that member attributes can also be specified. For example, a member can be identified as a key or a member can identify an alias. (*See* pg. 22, lines 3-13). Wang *et al.* merely discloses that the one-to-many relationships can be composed into two groups, privately owned and independent. Wherein, a privately owned relationship is one in which the target object is a dependent part of the source object and cannot exist on its own without the source object. (*See* col. 4, lines 29-36). Wang *et al.* does not disclose utilizing metadata associated with a database to generate an implementation neutral or implementation specific format that represents the precise database structure and data, as in applicants' claimed subject matter. Specifically, applicants' claimed subject matter defines the object schema in a declarative manner such that it is specified as information external to programming logic. In contrast, Wang *et al.* discloses managing object to relational, one-to-many mappings.

Ludwig *et al.* does not make up for the aforementioned deficiencies of Wotring *et al.* and Wang *et al.*, with respect to independent claim 1. Ludwig *et al.* discloses a database client/server

development system providing support for remote sessions with user-created application objects. When a user desires to create a remote-able object from a user object, the user assigns a proxy name or alias, thereby providing a mechanism to differentiate the real (local) version of the object from a remote version of that object. (*See* col. 3, lines 18-25).

As stated *supra*, applicants' claimed subject matter discloses a system for mapping object components to relational components that describes data classes. The members associated with each class to be persisted are identified. Furthermore, it should be appreciated that member attributes can also be specified. For example, a member can be identified as a key or a member can identify an alias. The alias is employed by a query language to identify the private member used to generate a query. The alias value points to a public member that is utilized instead of the associated private member in the text of a query. (*See* pg. 13, lines 10-15). Ludwig *et al.* merely discloses assigning a proxy name or alias to a remote-able object, this provides a mechanism to differentiate the real (local) version of the object from the remote version of that object. (*See* col. 9, line 65 – col. 10, line 5). Ludwig *et al.* does not disclose employing an alias by a query language to identify a private member, but merely uses a proxy name or alias to differentiate between a local version and a remote version of an object.

Chua does not make up for the aforementioned deficiencies of Wotring *et al.*, Wang *et al.* and Ludwig *et al.*, with respect to independent claim 1. Chua discloses a method and system for management and serving of multiple versions/releases of the same program from a single application server. The system utilizes one or more JNDI proxies situated between each client and an application server. The JNDI proxies allow a same public "service name" to be utilized by different clients to access to different programs/services on the application server. The JNDI proxies do this by translating the service name into a non-public "alias name" on behalf of the client. The alias name is a private name that the service provider who administers the application server understands and uses to locate the specific version of programs/services that the clients need. (*See* pg. 1, paragraph [0009]).

Applicants' claimed subject matter discloses member properties that include a hidden attribute that can be a Boolean value defining if there is a hidden member such as a field in a corresponding class. Hidden attributes can be employed for implicitly managed keys as well as time stamp and row version properties. Users don't want to complicate their object model with such properties, so they can be hidden members. Applicants' claimed subject matter provides for

hidden member storage and manages the hidden members in a transparent fashion. Users can choose to expose hidden members in the object model if they so desire by setting hidden = "false" or by not utilizing the hidden attribute. Chua merely discloses utilizing JNDI proxies to translate a service name into a non-public "alias name" on behalf of the client. Chua does not disclose providing for hidden member storage and managing the hidden members in a transparent fashion, as in applicants' claimed system.

Furthermore, independent claim 22 recites a method for producing an object schema comprising: *specifying classes to be persisted to a data store; identifying members of each class,...; specifying relationships between classes and persisting object data to the data store; utilizing an object schema, a relational schema and a mapping schema to map the object data to tables in the data store; utilizing metadata associated with the database to generate an implementation neutral or an implementation specific format that represents the data store structure; and* ***identifying a name of a member to be used as an alias to query a private member, the alias points to a public member that is to be utilized in place of the associated private member in text of a query; and defining a hidden member in a corresponding class and managing the hidden member in a transparent fashion***.

As stated *supra*, Wotring *et al.* merely discloses transforming data stored in relational format into a hierarchical format such as a markup language. The system provides for mapping each of the plurality of elements in the hierarchical data entity to information in a relational dataset, and transforming the relational dataset into corresponding mapped elements in the hierarchical data entity to form a hierarchical data structure. And, Wang *et al.* merely discloses that the one-to-many relationships can be composed into two groups, privately owned and independent. Wherein, a privately owned relationship is one in which the target object is a dependent part of the source object and cannot exist on its own without the source object. And, Ludwig *et al.* merely discloses assigning a proxy name or alias to a remote-able object, this provides a mechanism to differentiate the real (local) version of the object from the remote version of that object. And, Chua discloses a method and system for management and serving of multiple versions/releases of the same program from a single application server. The system utilizes one or more JNDI proxies situated between each client and an application server. Accordingly, Wotring *et al.*, Wang *et al.*, Ludwig *et al.* and Chua do not disclose providing for

hidden member storage and managing the hidden members in a transparent fashion, as in applicants' claimed method.

Furthermore, independent claim 34 recites a method for generating an object schema comprising: *receiving program code that describes one or more classes which define objects; describing members of each class,...; receiving input from a developer; generating an object schema to be employed to facilitate mapping object components from an object oriented program to tables in a relational database; providing a mapping schema that provides a mapping between the object schema and a relational schema, and* ***the relational schema utilizes metadata associated with the database to generate an implementation neutral or an implementation specific format that represents the database structure; identifying a name of a member to be used as an alias to query a private member, the alias points to a public member that is to be utilized in place of the associated private member in text of a query; and defining a hidden member in a corresponding class and managing the hidden member in a transparent fashion***.

As stated *supra*, Wotring *et al.* merely discloses mapping each of a plurality of elements in the hierarchical data entity to information in a relational dataset, and transforming the relational dataset into corresponding mapped elements in the hierarchical data entity to form a hierarchical data structure. And, Wang *et al.* merely discloses that the one-to-many relationships can be composed into two groups, privately owned and independent. And, Ludwig *et al.* merely discloses assigning a proxy name or alias to a remote-able object, this provides a mechanism to differentiate the real (local) version of the object from the remote version of that object. And, Chua discloses a method and system for management and serving of multiple versions/releases of the same program from a single application server. The system utilizes one or more JNDI proxies situated between each client and an application server. Accordingly, Wotring *et al.*, Wang *et al.*, Ludwig *et al.* and Chua do not disclose providing for hidden member storage and managing the hidden members in a transparent fashion, as in applicants' claimed method.

In view of the aforementioned deficiencies of the cited references, it is respectfully submitted that this rejection be withdrawn with respect to independent claims 1, 22 and 34 (which claims 3-4, 6-12, 23-25, 27-33, 35 and 37-39 depend respectively there from).

III. Rejection of Claims 13-14, 16-17, 19, and 21 Under 35 U.S.C. §103(a)

Claims 13-14, 16-17, 19, and 21 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Wang *et al.* (US 6,907,433), in view of Bigus *et al.* (US 7,136,843), further in view of Ludwig *et al.* (US 6,006,230), and further in view of Chua (US 2005/0210124). It is respectfully requested that this rejection should be withdrawn for at least the following reasons. Wang *et al.*, Bigus *et al.*, Ludwig *et al.* and Chua, individually or in combination, do not teach or suggest each and every element as set forth in the subject claims.

Applicants' claimed subject matter relates to a system and method to facilitate mapping between disparate domain models such as an object oriented programming model and a relational database model. Specifically, the system discloses an object schema component that provides a mechanism for providing a bridge between a transient graph of objects and a related persistent data store. In particular, the object schema provides metadata describing a given set of classes in addition to a program assembly containing type definitions. The metadata can subsequently be utilized by a mapping system to translate relational data to and from user objects during a materialization or persistence process.

Independent claim 13 recites: an object schema generation system comprising: *a code reader component adapted to read or retrieve code from an object-oriented program or set of programs,...; wherein the mapping schema provides the mapping between the object schema and the relational schema, and the relational schema utilizes metadata associated with the data store to generate an implementation specific format that represents the data store structure; and* ***wherein properties of the members of classes include an alias attribute that is employed by a query language to identify a private member used to generate a query, the alias points to a public member that is to be utilized in place of the associated private member in text of a query; wherein the object schema generation component utilizes rule based artificial intelligence to provide heuristics necessary to build the schema; and wherein properties of the members of classes include a hidden attribute that defines if there is a hidden member in a corresponding class and manages the hidden member in a transparent fashion***. The cited references do not expressly or inherently disclose the aforementioned novel aspects of applicants' claimed subject matter as recited in the subject claims.

Wang *et al.* discloses a system for managing object to relational, one-to-many mapping. The system uses mapping of meta-data to generate instructions to manipulate target objects and

relationships in a relational database. The mapping of meta-data contains information as to how object classes of the object model map to tables in the database and how relationships map to foreign keys. (*See* col. 2, lines 3-25).

As stated *supra*, applicants' claimed subject matter discloses a system for mapping object components to relational components that describes data classes. The members associated with each class to be persisted are identified. Members can include class fields and properties. Members can be compound members comprising at least one field or property and another compound member. Thus, a compound member can be an array. Furthermore, it should be appreciated that member attributes can also be specified. For example, a member can be identified as a key or a member can identify an alias. (*See* pg. 22, lines 3-13). Wang *et al.* merely discloses that the one-to-many relationships can be composed into two groups, privately owned and independent. Wherein, a privately owned relationship is one in which the target object is a dependent part of the source object and cannot exist on its own without the source object. (*See* col. 4, lines 29-36). Wang *et al.* does not disclose identifying a name of a member to be used as an alias to query a private number.

Bigus *et al.* does not make up for the aforementioned deficiencies of Wang *et al.*, with respect to independent claim 13. Bigus *et al.* discloses a plurality of machine reasoning modules or inference engines that are processed against a single rule-based knowledge representation (rule language). This allows a user to maintain a single repository of domain knowledge for use by the plurality of inference engines. (*See* col. 2, lines 7-15).

As stated *supra*, applicants' claimed subject matter discloses a system for mapping object components to relational components that describes data classes. The members associated with each class to be persisted are identified. Furthermore, it should be appreciated that member attributes can also be specified. For example, a member can be identified as a key or a member can identify an alias. The alias is employed by a query language to identify the private member used to generate a query. The alias value points to a public member that is utilized instead of the associated private member in the text of a query. (*See* pg. 13, lines 10-15). Bigus *et al.* merely discloses the utilization of a rule based artificial intelligence and does not disclose employing an alias by a query language to identify a private member.

Ludwig *et al.* does not make up for the aforementioned deficiencies of Wang *et al.* and Bigus *et al.* Ludwig *et al.* discloses a database client/server development system providing

support for remote sessions with user-created application objects. When a user desires to create a remote-able object from a user object, the user assigns a proxy name or alias, thereby providing a mechanism to differentiate the real (local) version of the object from a remote version of that object. (*See* col. 3, lines 18-25).

As stated *supra*, applicants' claimed subject matter discloses a system for mapping object components to relational components that describes data classes. The members associated with each class to be persisted are identified. Furthermore, it should be appreciated that member attributes can also be specified. For example, a member can be identified as a key or a member can identify an alias. The alias is employed by a query language to identify the private member used to generate a query. The alias value points to a public member that is utilized instead of the associated private member in the text of a query. (*See* pg. 13, lines 10-15). Ludwig *et al.* merely discloses assigning a proxy name or alias to a remote-able object, this provides a mechanism to differentiate the real (local) version of the object from the remote version of that object. (*See* col. 9, line 65 – col. 10, line 5). Ludwig *et al.* does not disclose employing an alias by a query language to identify a private member, but merely uses a proxy name or alias to differentiate between a local version and a remote version of an object.

Chua does not make up for the aforementioned deficiencies of Wang *et al.*, Bigus *et al.* and Ludwig *et al.* Chua discloses a method and system for management and serving of multiple versions/releases of the same program from a single application server. The system utilizes one or more JNDI proxies situated between each client and an application server. The JNDI proxies allow a same public "service name" to be utilized by different clients to access to different programs/services on the application server. The JNDI proxies do this by translating the service name into a non-public "alias name" on behalf of the client. The alias name is a private name that the service provider who administers the application server understands and uses to locate the specific version of programs/services that the clients need. (*See* pg. 1, paragraph [0009]).

Applicants' claimed subject matter discloses member properties that include a hidden attribute that can be a Boolean value defining if there is a hidden member such as a field in a corresponding class. Hidden attributes can be employed for implicitly managed keys as well as time stamp and row version properties. Users don't want to complicate their object model with such properties, so they can be hidden members. Applicants' claimed subject matter provides for hidden member storage and manages the hidden members in a transparent fashion. Users can

choose to expose hidden members in the object model if they so desire by setting hidden = "false" or by not utilizing the hidden attribute. Chua merely discloses utilizing JNDI proxies to translate a service name into a non-public "alias name" on behalf of the client. Chua does not disclose providing for hidden member storage and managing the hidden members in a transparent fashion, as in applicants' claimed system.

In view of the aforementioned deficiencies of the cited references, it is respectfully submitted that this rejection be withdrawn with respect to independent claim 13 (and claims 14, 16-17, 19 and 21 which depend there from). Accordingly, it is respectfully requested that these claims be deemed allowable.

CONCLUSION

The present application is believed to be in condition for allowance in view of the above comments and amendments. A prompt action to such end is earnestly solicited.

In the event any fees are due in connection with this document, the Commissioner is authorized to charge those fees to Deposit Account No. 50-1063 [MSFTP567US].

Should the Examiner believe a telephone interview would be helpful to expedite favorable prosecution, the Examiner is invited to contact applicants' undersigned representative at the telephone number below.

Respectfully submitted,

AMIN, TUROCY & CALVIN, LLP

/Marisa J. Zink/
Marisa J. Zink
Reg. No. 48,064

AMIN, TUROCY & CALVIN, LLP
57TH Floor, Key Tower
127 Public Square
Cleveland, Ohio 44114
Telephone (216) 696-8730
Facsimile (216) 696-8731